

**Listing of Claims:**

1                   1.       (previously presented) A method for servicing interrupts generated by a  
2 plurality of co-processors included in a multiprocessor subsystem, the method comprising the  
3 acts of:  
4                   in response to a detected interrupt, determining whether the detected interrupt was  
5 generated by one of the plurality of co-processors of the multiprocessor subsystem; and  
6                   in the event that the detected interrupt was generated by one of the plurality of co-  
7 processors, scheduling execution of a deferred servicing procedure,  
8                   wherein during execution the deferred servicing procedure services a plurality of  
9 pending interrupts generated by two or more of the plurality of co-processors, including the  
10 detected interrupt.

1                   2.       (previously presented) The method of claim 1, wherein during execution  
2 the deferred servicing procedure services all pending interrupts from all of the plurality of co-  
3 processors.

1                   3.       (previously presented) The method of claim 1, wherein the plurality of  
2 pending interrupts serviced by the deferred servicing procedure includes a second interrupt  
3 generated by the one of the plurality of co-processors that generated the detected interrupt.

1                   4.       (previously presented) The method of claim 1, wherein the plurality of  
2 pending interrupts serviced by the deferred servicing procedure includes a second interrupt  
3 generated by one of the plurality of co-processors other than the one that generated the detected  
4 interrupt.

1                   5.       (original) The method of claim 1, wherein the act of determining whether  
2 the detected interrupt was generated by one of the plurality of co-processors includes the acts of:  
3                   selecting one of the plurality of co-processors as a current co-processor; and  
4                   reading a value stored in an interrupt register of the current co-processor.

1                   6.       (original) The method of claim 5, wherein in the event that the value  
2 stored in the interrupt register does not indicate an interrupt, a different one of the co-processors  
3 is selected and the act of reading is repeated.

1                   7.       (original) The method of claim 5, wherein the act of reading includes:  
2                   updating a private register mapping to enable access to the interrupt register of the  
3 current co-processor,  
4                   wherein the private mapping is not used by the deferred servicing procedure.

1                   8.       (original) The method of claim 1, further comprising the act of disabling  
2 further interrupts from the plurality of co-processors in the event that the detected interrupt was  
3 generated by one of the plurality of co-processors,  
4                   wherein during execution the deferred servicing procedure re-enables interrupts  
5 from the plurality of co-processors.

1                   9.       (original) The method of claim 8, wherein the act of disabling further  
2 interrupts is performed at a critical priority level.

1                   10.      (original) The method of claim 1, wherein the act of determining whether  
2 the detected interrupt was generated by one of the plurality of co-processors is performed at a  
3 critical priority level.

1                   11.      (original) The method of claim 10, wherein the act of scheduling  
2 execution of the deferred servicing procedure is performed at the critical priority level.

1                   12.      (original) The method of claim 11, wherein the act of scheduling  
2 execution of the deferred servicing procedure includes setting a second priority level for the  
3 deferred servicing procedure, wherein the second priority level is lower than the critical priority  
4 level.

1                   13.     (original) The method of claim 1, wherein the multiprocessor subsystem is  
2 a graphics processing subsystem.

1                   14.     (original) A computer system comprising:  
2                   a multiprocessor subsystem including a plurality of co-processors for processing  
3 data, wherein each of the co-processors is configured to generate interrupts; and  
4                   a driver module configured to control operation of the multiprocessor subsystem,  
5 the driver module including:  
6                   a schedulable servicing module configured to detect and service all pending  
7 interrupts from all of the co-processors when activated; and  
8                   an interrupt detection module configured to schedule the servicing module for  
9 activation in the event of an interrupt from any one of the plurality of co-processors.

1                   15.     (original) The system of claim 14, wherein the interrupt detection module  
2 is further configured to be activated by a central processing unit of the computer system in  
3 response to an interrupt signal.

1                   16.     (original) The system of claim 14, wherein the interrupt detection module  
2 is further configured to disable further interrupts from all of the co-processors in the event of an  
3 interrupt from any one of the co-processors, and wherein the servicing module is further  
4 configured to re-enable further interrupts from all of the co-processors.

1                   17.     (original) The system of claim 14, wherein the interrupt detection module  
2 is further configured to operate at a critical priority level and the servicing module is further  
3 configured to operate at a second priority level lower than the critical priority level.

1                   18.     (original) The system of claim 14, wherein the multiprocessor subsystem  
2 is configured for graphics processing.

1                   19.     (original) The system of claim 14, wherein each of the plurality of co-  
2 processors includes an interrupt register configured to indicate an interrupt, and wherein the

3 interrupt detection module is further configured to determine whether one of the plurality of co-  
4 processors generated an interrupt by accessing the respective interrupt registers of the co-  
5 processors.

1 20. (previously presented) The system of claim 19, wherein the interrupt  
2 detection module is further configured to maintain a private mapping for accessing the respective  
3 interrupt registers, the private mapping being used exclusively by the interrupt detection module.

1 21. (previously presented) A computer program product comprising:  
2 a computer readable storage medium encoded with program code, the program  
3 code including:  
4 program code for determining, in response to a detected interrupt, whether the  
5 detected interrupt was generated by one of the plurality of co-processors of the multiprocessor  
6 subsystem;  
7 program code for scheduling a deferred servicing procedure in the event that the  
8 detected interrupt was generated by one of the plurality of co-processors; and  
9 program code for performing the deferred servicing procedure, wherein the  
10 program code for performing the deferred servicing procedure includes program code for  
11 servicing a plurality of pending interrupts from two or more of the plurality of processors,  
12 including the detected interrupt.

1 22. (previously presented) The method of claim 1, further comprising:  
2 determining whether interrupts from the plurality of co-processors are enabled;  
3 and  
4 if the interrupts from the plurality of co-processors are not enabled, exiting  
5 without performing further processing.

1 23. (New) The method of claim 2, wherein servicing all pending interrupts  
2 from all of the plurality of co-processors comprises:

3                   loading by the deferred servicing procedure a mapping to the registers of a next  
4 co-processor to be serviced; and  
5                   servicing any and all pending interrupts stored in the registers of the next  
6 co-processor.